Adaptive Feature Selection for End-to-End Speech Translation

Adaptive Feature Selection for End-to-End Speech Translation

Biao Zhang[1] **Ivan Titov**[1,2] **Barry Haddow**[1] **Rico Sennrich**[3,1]

[1]School of Informatics, University of Edinburgh
[2]ILLC, University of Amsterdam
[3]Department of Computational Linguistics, University of Zurich
B.Zhang@ed.ac.uk, {ititov,bhaddow}@inf.ed.ac.uk, sennrich@cl.uzh.ch

Abstract

Information in speech signals is not evenly distributed, making it an additional challenge for end-to-end (E2E) speech translation (ST) to learn to focus on informative features. In this paper, we propose adaptive feature selection (AFS) for encoder-decoder based E2E ST. We first pre-train an ASR encoder and apply AFS to dynamically estimate the importance of each encoded speech feature to ASR. A ST encoder, stacked on top of the ASR encoder, then receives the filtered features from the (frozen) ASR encoder. We take $\mathcal{L}_0$DROP (Zhang et al., 2020) as the backbone for AFS, and adapt it to sparsify speech features with respect to both temporal and feature dimensions. Results on LibriSpeech En-Fr and MuST-C benchmarks show that AFS facilitates learning of ST by pruning out ∼84% temporal features, yielding an average translation gain of ∼1.3–1.6 BLEU and a decoding speedup of ∼1.4×. In particular, AFS reduces the performance gap compared to the cascade baseline, and outperforms it on LibriSpeech En-Fr with a BLEU score of 18.56 (without data augmentation).[1]

1 Introduction

End-to-end (E2E) speech translation (ST), a paradigm that directly maps audio to a foreign text, has been gaining popularity recently (Duong et al., 2016; Bérard et al., 2016; Bansal et al., 2018; Di Gangi et al., 2019; Wang et al., 2019). Based on the attentional encoder-decoder framework (Bahdanau et al., 2015), it optimizes model parameters under direct translation supervision. This end-to-end paradigm avoids the problem of error propagation that is inherent in cascade models where an automatic speech recognition (ASR) model and a machine translation (MT) model are chained together. Nonetheless, previous work still reports that E2E ST delivers inferior performance compared to cascade methods (Niehues et al., 2019).

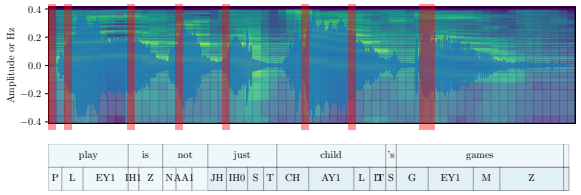


Figure 1: Example illustrating our motivation. We plot the amplitude and frequency spectrum of an audio segment (top), paired with its time-aligned words and phonemes (bottom). Information inside an audio stream is not uniformly distributed. We propose to dynamically capture speech features corresponding to informative signals (red rectangles) to improve ST.

We study one reason for the difficulty of training E2E ST models, namely the uneven spread of information in the speech signal, as visualized in Figure 1, and the consequent difficulty of extracting informative features. Features corresponding to uninformative signals, such as pauses or noise, increase the input length and bring in unmanageable noise for ST. This increases the difficulty of learning (Zhang et al., 2019b; Na et al., 2019) and reduces translation performance.

In this paper, we propose adaptive feature selection (AFS) for ST to explicitly eliminate uninformative features. Figure 2 shows the overall architecture. We employ a pretrained ASR encoder to induce contextual speech features, followed by an ST encoder bridging the gap between speech and translation modalities. AFS is inserted in-between them to select a subset of features for ST encoding (see red rectangles in Figure 1). To ensure that the selected features are well-aligned to transcriptions, we pretrain AFS on ASR. AFS estimates the informativeness of each feature through a parameterized gate, and encourages the dropping of

[1]We release our source code at https://github.com/bzhangGo/zero.

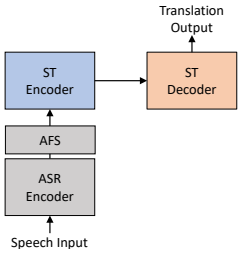


Figure 2: Overview of our E2E ST model. AFS is inserted between the ST encoder (blue) and a pretrained ASR encoder (gray) to filter speech features for translation. We pretrain AFS jointly with ASR and freeze it during ST training.

features (pushing the gate to 0) that contribute little to ASR. An underlying assumption is that features irrelevant for ASR are also unimportant for ST.

We base AFS on $\mathcal{L}_0$DROP (Zhang et al., 2020), a sparsity-inducing method for encoder-decoder models, and extend it to sparsify speech features. The acoustic input of speech signals involves two dimensions: *temporal* and *feature*, where the latter one describes the spectrum extracted from time frames. Accordingly, we adapt $\mathcal{L}_0$DROP to sparsify encoder states along temporal and feature dimensions but using different gating networks. In contrast to (Zhang et al., 2020), who focus on efficiency and report a trade-off between sparsity and quality for MT and summarization, we find that sparsity also improves translation quality for ST.

We conduct extensive experiments with Transformer (Vaswani et al., 2017) on LibriSpeech En-Fr and MuST-C speech translation tasks, covering 8 different language pairs. Results show that AFS only retains about 16% of temporal speech features, revealing heavy redundancy in speech encodings and yielding a decoding speedup of $\sim$1.4$\times$. AFS eases model convergence, and improves the translation quality by $\sim$1.3–1.6 BLEU, surpassing several strong baselines. Specifically, without data augmentation, AFS narrows the performance gap against the cascade approach, and outperforms it on LibriSpeech En-Fr by 0.29 BLEU, reaching 18.56. We compare against fixed-rate feature selection and a simple CNN, confirming that our adaptive feature selection offers better translation quality.

Our work demonstrates that E2E ST suffers from redundant speech features, with sparsification bringing significant performance improvements. The E2E ST task offers new opportunities for follow-up research in sparse models to deliver performance gains, apart from enhancing efficiency and/or interpretability.

2 Background: $\mathcal{L}_0$DROP

$\mathcal{L}_0$DROP provides a selective mechanism for encoder-decoder models which encourages removing uninformative encoder outputs via a sparsity-inducing objective (Zhang et al., 2020). Given a source sequence $X = \{x_1, x_2, \ldots, x_n\}$, $\mathcal{L}_0$DROP assigns each encoded source state $\mathbf{x}_i \in \mathbb{R}^d$ with a scalar gate $g_i \in [0, 1]$ as follows:

$$\mathcal{L}_0\text{DROP}(\mathbf{x}_i) = g_i \mathbf{x}_i, \quad (1)$$

$$\text{with } g_i \sim \text{HardConcrete}(\alpha_i, \beta, \epsilon), \quad (2)$$

where $\alpha_i, \beta, \epsilon$ are hyperparameters of the hard concrete distribution (HardConcrete) (Louizos et al., 2018).

Note that the hyperparameter α_i is crucial to HardConcrete as it directly governs its shape. We associate α_i with $\mathbf{x}_i$ through a gating network:

$$\log \alpha_i = \mathbf{x}_i^T \cdot \mathbf{w}, \quad (3)$$

Thus, $\mathcal{L}_0$DROP can schedule HardConcrete via α_i to put more probability mass at either 0 (i.e $g_i \to 0$) or 1 (i.e. $g_i \to 1$). $\mathbf{w} \in \mathbb{R}^d$ is a trainable parameter. Intuitively, $\mathcal{L}_0$DROP controls the openness of gate g_i via α_i so as to determine whether to remove ($g_i = 0$) or retain ($g_i = 1$) the state $\mathbf{x}_i$.

$\mathcal{L}_0$DROP enforces sparsity by pushing the probability mass of HardConcrete towards 0, according to the following penalty term:

$$\mathcal{L}_0(X) = \sum_{i=1}^{n} 1 - p(g_i = 0 | \alpha_i, \beta, \epsilon). \quad (4)$$

By sampling g_i with reparameterization (Kingma and Welling, 2013), $\mathcal{L}_0$DROP is fully differentiable and optimized with an upper bound on the objective: $\mathcal{L}_{\text{MLE}} + \lambda \mathcal{L}_0(X)$, where λ is a hyperparameter affecting the degree of sparsity – a larger λ enforces more gates near 0 – and $\mathcal{L}_{\text{MLE}}$ denotes the maximum likelihood loss. An estimation of the expected value of g_i is used during inference. Zhang et al. (2020) applied $\mathcal{L}_0$DROP to prune encoder outputs for MT and summarization tasks; we adapt it to E2E ST. Sparse stochastic gates and $\mathcal{L}_0$ relaxations were also by Bastings et al. (2019) to construct interpretable classifiers, i.e. models that can reveal which tokens they rely on when making a prediction.

3 Adaptive Feature Selection

One difficulty with applying encoder-decoder models to E2E ST is deciding how to encode speech

signals. In contrast to text where word boundaries can be easily identified, the spectrum features of speech are continuous, varying remarkably across different speakers for the same transcript. In addition, redundant information, like pauses in-between neighbouring words, can be of arbitrary duration at any position as shown in Figure 1, while contributing little to translation. This increases the burden and occupies the capacity of ST encoder, leading to inferior performance (Duong et al., 2016; Bérard et al., 2016). Rather than developing complex encoder architectures, we resort to feature selection to explicitly clear out those uninformative speech features.

Figure 2 gives an overview of our model. We use a pretrained and frozen ASR encoder to extract contextual speech features, and collect the informative ones from them via AFS before transmission to the ST encoder. AFS drops pauses, noise and other uninformative features and retains features that are relevant for ASR. We speculate that these retained features are also the most relevant for ST, and that the sparser representation simplifies the learning problem for ST, for example the learning of attention strength between encoder states and target language (sub)words. Given a training tuple (audio, source transcription, translation), denoted as (X, Y, Z) respectively,[2] we outline the overall framework below, including three steps:

E2E ST with AFS

1. Train ASR model with the following objective and model architecture until convergence:

$$\mathcal{L}^{\text{ASR}} = \eta\mathcal{L}_{\text{MLE}}(Y|X) + \gamma\mathcal{L}_{\text{CTC}}(Y|X), \quad (5)$$

$$\mathcal{M}^{\text{ASR}} = D^{\text{ASR}}\left(Y, E^{\text{ASR}}(X)\right). \quad (6)$$

2. Finetune ASR model with AFS for m steps:

$$\mathcal{L}^{\text{AFS}} = \mathcal{L}_{\text{MLE}}(Y|X) + \lambda\mathcal{L}_0(X), \quad (7)$$

$$\mathcal{M}^{\text{AFS}} = D^{\text{ASR}}\left(Y, F\left(E^{\text{ASR}}(X)\right)\right). \quad (8)$$

3. Train ST model with pretrained and frozen ASR and AFS submodules until convergence:

$$\mathcal{L}^{\text{ST}} = \mathcal{L}_{\text{MLE}}(Z|X), \quad (9)$$

$$\mathcal{M}^{\text{ST}} = D^{\text{ST}}\left(Z, E^{\text{ST}}\left(\overline{FE}^{\text{ASR}}(X)\right)\right). \quad (10)$$

We handle both ASR and ST as sequence-to-sequence problem with encoder-decoder models. We use $E^*(\cdot)$ and $D^*(\cdot,\cdot)$ to denote the corresponding encoder and decoder respectively. $F(\cdot)$ denotes the AFS approach, and $\overline{FE}$ means freezing the ASR encoder and the AFS module during training. Note that our framework puts no constraint on the architecture of the encoder and decoder in any task, although we adopt the multi-head dot-product attention network (Vaswani et al., 2017) for our experiments.

ASR Pretraining The ASR model $\mathcal{M}^{\text{ASR}}$ (Eq. 6) directly maps an audio input to its transcription. To improve speech encoding, we apply logarithmic penalty on attention to enforce short-range dependency (Di Gangi et al., 2019) and use trainable positional embedding with a maximum length of 2048. Apart from $\mathcal{L}_{\text{MLE}}$, we augment the training objective with the connectionist temporal classification (Graves et al., 2006, CTC) loss $\mathcal{L}_{\text{CTC}}$ as in Eq. 5. Note $\eta = 1 - \gamma$. The CTC loss is applied to the encoder outputs, guiding them to align with their corresponding transcription (sub)words and improving the encoder's robustness (Karita et al., 2019). Following previous work (Karita et al., 2019; Wang et al., 2020), we set γ to 0.3.

AFS Finetuning This stage aims at using AFS to dynamically pick out the subset of ASR encoder outputs that are most relevant for ASR performance (see red rectangles in Figure 1). We follow Zhang et al. (2020) and place AFS in-between ASR encoder and decoder during finetuning (see $F(\cdot)$ in $\mathcal{M}^{\text{AFS}}$, Eq. 8). We exclude the CTC loss in the training objective (Eq. 7) to relax the alignment constraint and increase the flexibility of feature adaptation. We use $\mathcal{L}_0$DROP for AFS in two ways.

AFS^t The direct application of $\mathcal{L}_0$DROP on ASR encoder results in AFS^t, sparsifying encodings along the temporal dimension $\{\mathbf{x}_i\}_{i=1}^n$:

$$\begin{aligned} F^t(\mathbf{x}_i) &= \text{AFS}^t(\mathbf{x}_i) = g_i^t\mathbf{x}_i, \\ \text{with}\quad \log\alpha_i^t &= \mathbf{x}_i^T\cdot\mathbf{w}^t, \\ g_i^t &\sim \text{HardConcrete}(\alpha_i^t, \beta, \epsilon), \end{aligned} \quad (11)$$

where α_i^t is a positive scalar powered by a simple linear gating layer, and $\mathbf{w}^t \in \mathbb{R}^d$ is a trainable parameter of dimension d. $\mathbf{g}^t$ is the temporal gate. The sparsity penalty of AFS^t follows Eq. 4:

$$\mathcal{L}_0^t(X) = \sum_{i=1}^{n} 1 - p(g_i^t = 0|\alpha_i^t, \beta, \epsilon). \quad (12)$$

AFS^{t,f} In contrast to text processing, speech processing often extracts spectrum from overlapping

[2] Note that our model only requires pair-wise training corpora, (X, Y) for ASR, and (X, Z) for ST.

time frames to form the acoustic input, similar to the word embedding. As each encoded speech feature contains temporal information, it is reasonable to extend AFS^t to $\text{AFS}^{t,f}$, including sparsification along the feature dimension $\{\mathbf{x}_{i,j}\}_{j=1}^{d}$:

$$\begin{aligned} F^{t,f}(\mathbf{x}_i) &= \text{AFS}^{t,f}(\mathbf{x}_i) = g_i^t \mathbf{x}_i \odot \mathbf{g}^f, \\ \text{with} \quad \log \boldsymbol{\alpha}^f &= \mathbf{w}^f, \\ g_j^f &\sim \text{HardConcrete}(\alpha_j^f, \beta, \epsilon), \end{aligned} \tag{13}$$

where $\boldsymbol{\alpha}^f \in \mathbb{R}^d$ estimates the weights of each feature, dominated by an input-independent gating model with trainable parameter $\mathbf{w}^f \in \mathbb{R}^d$.[3] $\mathbf{g}^f$ is the feature gate. Note that $\boldsymbol{\alpha}^f$ is shared for all time steps. $\odot$ denotes element-wise multiplication. $\text{AFS}^{t,f}$ reuses g_i^t-relevant submodules in Eq. 11, and extends the sparsity penalty $\mathcal{L}_0^t$ in Eq. 12 as follows:

$$\mathcal{L}_0^{t,f}(X) = \mathcal{L}_0^t + \sum_{j=1}^{d} 1 - p(g_j^f = 0 | \alpha_j^f, \beta, \epsilon). \tag{14}$$

We perform the finetuning by replacing $(F, \mathcal{L}_0)$ in Eq. (8-7) with either AFS^t $(F^t, \mathcal{L}_0^t)$ or $\text{AFS}^{t,f}$ $(F^{t,f}, \mathcal{L}_0^{t,f})$ for extra m steps. We compare these two variants in our experiments.

E2E ST Training We treat the pretrained ASR and AFS model as a speech feature extractor, and freeze them during ST training. We gather the speech features emitted by the ASR encoder that correspond to $g_i^t > 0$, and pass them similarly as done with word embeddings to the ST encoder. We employ sinusoidal positional encoding to distinguish features at different positions. Except for the input to the ST encoder, our E2E ST follows the standard encoder-decoder translation model ($\mathcal{M}^{\text{ST}}$ in Eq. 10) and is optimized with $\mathcal{L}_{\text{MLE}}$ alone as in Eq. 9. Intuitively, AFS bridges the gap between ASR output and MT input by selecting transcript-aligned speech features.

4 Experiments

Datasets and Preprocessing We experiment with two benchmarks: the Augmented LibriSpeech dataset (LibriSpeech En-Fr) (Kocabiyikoglu et al., 2018) and the multilingual MuST-C dataset (MuST-C) (Di Gangi et al., 2019). LibriSpeech En-Fr is collected by aligning e-books in French with English utterances of LibriSpeech, further augmented with French translations offered by Google Translate. We use the 100 hours clean training set for training, including 47K utterances to train ASR models and double the size for ST models after concatenation with the Google translations. We report results on the test set (2048 utterances) using models selected on the dev set (1071 utterances). MuST-C is built from English TED talks, covering 8 translation directions: English to German (De), Spanish (Es), French (Fr), Italian (It), Dutch (Nl), Portuguese (Pt), Romanian (Ro) and Russian (Ru). We train ASR and ST models on the given training set, containing ∼452 hours with ∼252K utterances on average for each translation pair. We adopt the given dev set for model selection and report results on the common test set, whose size ranges from 2502 (Es) to 2641 (De) utterances.

For all datasets, we extract 40-dimensional log-Mel filterbanks with a step size of 10ms and window size of 25ms as the acoustic features. We expand these features with their first and second-order derivatives, and stabilize them using mean subtraction and variance normalization. We stack the features corresponding to three consecutive frames without overlapping to the left, resulting in the final 360-dimensional acoustic input. For transcriptions and translations, we tokenize and truecase all the text using Moses scripts (Koehn et al., 2007). We train subword models (Sennrich et al., 2016) on each dataset with a joint vocabulary size of 16K to handle rare words, and share the model for ASR, MT and ST. We train all models without removing punctuation.

Model Settings and Baselines We adopt the Transformer architecture (Vaswani et al., 2017) for all tasks, including $\mathcal{M}^{\text{ASR}}$ (Eq. 6), $\mathcal{M}^{\text{AFS}}$ (Eq. 8) and $\mathcal{M}^{\text{ST}}$ (Eq. 10). The encoder and decoder consist of 6 identical layers, each including a self-attention sublayer, a cross-attention sublayer (decoder alone) and a feedforward sublayer. We employ the base setting for experiments: hidden size $d = 512$, attention head 8 and feedforward size 2048. We schedule learning rate via Adam ($\beta_1 = 0.9, \beta_2 = 0.98$) (Kingma and Ba, 2015), paired with a warmup step of 4K. We apply dropout to attention weights and residual connections with a rate of 0.1 and 0.2 respectively, and also add label smoothing of 0.1 to handle overfitting. We train all models with a maximum step size of 30K and a

[3] Other candidate gating models, like linear mapping upon mean-pooled encoder outputs, delivered worse performance in our preliminary experiments.

minibatch size of around 25K target subwords. We average the last 5 checkpoints for evaluation. We use beam search for decoding, and set the beam size and length penalty to 4 and 0.6, respectively. We set $\epsilon = -0.1$, and $\beta = 2/3$ for AFS following Louizos et al. (2018), and finetune AFS for an additional $m = 5$K steps. We evaluate translation quality with tokenized case-sensitive BLEU (Papineni et al., 2002), and report WER for ASR performance without punctuation.

We compare our models with four baselines:

ST: A vanilla Transformer-based E2E ST model of 6 encoder and decoder layers. Logarithmic attention penalty (Di Gangi et al., 2019) is used to improve the encoder.

ST + ASR-PT: We perform the ASR pretraining (ASR-PT) for E2E ST. This is the same model as ours (Figure 2) but without AFS finetuning.

Cascade: We first transcribe the speech input using an ASR model, and then passes the results on to an MT model. We also use the logarithmic attention penalty (Di Gangi et al., 2019) for the ASR encoder.

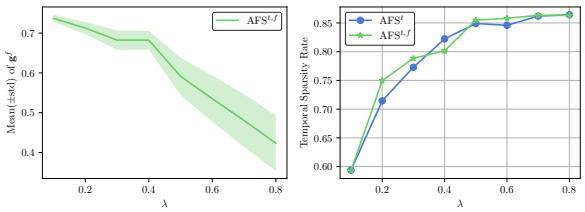
ST + Fixed Rate: Instead of dynamically selecting features, we replace AFS with subsampling at a fixed rate: we extract the speech encodings after every k positions.

Besides, we offer another baseline, **ST + CNN**, for comparison on MuST-C En-De: we replace the fixed-rate subsampling with a one-layer 1D depth-separable convolution, where the output dimension is set to 512, the kernel size over temporal dimension is set to 5 and the stride is set to 6. In this way, the ASR encoder features will be compressed to around 1/6 features, a similar ratio to the fixed-rate subsampling.

4.1 Results on MuST-C En-De

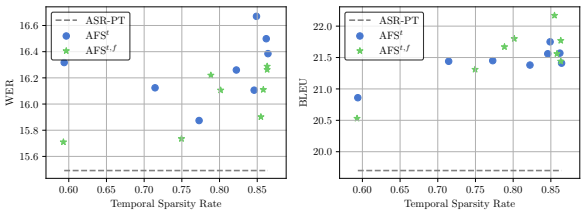
We perform a thorough study on MuST-C En-De. With AFS, the first question is its feasibility. We start by analyzing the degree of sparsity in speech features (i.e. sparsity rate) yielded by AFS, focusing on the temporal sparsity rate $\#\{g_i^t{=}0\}/n$ and the feature sparsity rate $\#\{g_j^f{=}0\}/d$. To obtain different rates, we vary the hyperparameter λ in Eq. 7 in a range of $[0.1, 0.8]$ with a step size 0.1.

Results in Figure 3 show that large amounts of encoded speech features ($> 59\%$) can be easily pruned out, revealing heavy inner-speech redundancy. Both AFS^t and AFS^{t,f} drop $\sim$60% temporal features with λ of 0.1, and this number increases



(a) Feature Gate Value (b) Temporal Sparsity Rate

Figure 3: Feature gate value and temporal sparsity rate as a function of λ on MuST-C En-De dev set. Larger λ decreases the gate value of $\mathbf{g}^f$ but without dropping any neurons, i.e. feature sparsity rate 0%. By contrast, speech features are of high redundancy along temporal dimension, easily inducing high sparsity rate of $\sim$85%.



(a) ASR (b) ST

Figure 4: ASR (WER↓) and ST (BLEU↑) performance as a function of *temporal* sparsity rate on MuST-C En-De dev set. Pruning out $\sim$85% temporal speech features largely improves translation quality and retains $\sim$95% ASR accuracy.

to $> 85\%$ when $\lambda \geq 0.5$ (Figure 3b), remarkably surpassing the sparsity rate reported by Zhang et al. (2020) on text summarization (71.5%). In contrast to rich temporal sparsification, we get a feature sparsity rate of 0, regardless of λ's value, although increasing λ decreases $\mathbf{g}^f$ (Figure 3a). This suggests that selecting neurons from the feature dimension is harder. Rather than filtering neurons, the feature gate $\mathbf{g}^f$ acts more like a weighting mechanism on them. In the rest of the paper, we use *sparsity rate* for the temporal sparsity rate.

We continue to explore the impact of varied sparsity rates on the ASR and ST performance. Figure 4 shows their correlation. We observe that AFS slightly degenerates ASR accuracy (Figure 4a), but still retains $\sim$95% accuracy on average; AFS^{t,f} often performs better than AFS^t with similar sparsity rate. The fact that only 15% speech features successfully support 95% ASR accuracy proves the informativeness of these selected features. These findings echo with (Zhang et al., 2020), where they observe a trade-off between sparsity and quality.

However, when AFS is applied to ST, we find consistent improvements to translation quality by > 0.8 BLEU, shown in Figure 4b. Translation quality on the development set peaks at 22.17 BLEU

Model	BLEU↑	Speedup↑
MT	29.69	-
Cascade	22.52	1.06×
ST	17.44	0.87×
ST + ASR-PT	20.67	1.00×
ST + CNN	20.64	1.31×
ST + Fixed Rate ($k = 6$)	21.14 (83.3%)	1.42×
ST + Fixed Rate ($k = 7$)	20.87 (85.7%)	1.43×
ST + AFS^t	21.57 (84.4%)	1.38×
ST + AFS^{t,f}	22.38 (85.1%)	1.37×

Table 1: BLEU↑ and speedup↑ on MuST-C En-De test set. $\lambda = 0.5$. We evaluate the speedup on GeForce GTX 1080 Ti with a decoding batch size of 16, and report average results over 3 runs. Numbers in parentheses are the sparsity rate.

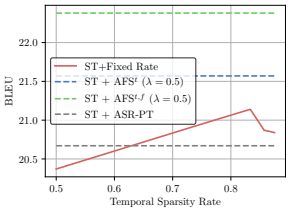


Figure 5: Impact of k in fixed-rate subsampling on ST performance on MuST-C En-De test set. Sparsity rate: $^{k-1}/_{k}$. This subsampling underperforms AFS, and degenerates the ST performance at suboptimal rates.

achieved by AFS^{t,f} with a sparsity rate of 85.5%. We set $\lambda = 0.5$ (corresponding to sparsity rate of ∼85%) for all other experiments, since AFS^t and AFS^{t,f} reach their optimal result at this point.

We summarize the test results in Table 1, where we set $k = 6$ or $k = 7$ for *ST+Fixed Rate* with a sparsity rate of around 85% inspired by our above analysis. Our vanilla ST model yields a BLEU score of 17.44; pretraining on ASR further enhances the performance to 20.67, significantly outperforming the results of Di Gangi et al. (2019) by 3.37 BLEU. This also suggests the importance of speech encoder pretraining (Di Gangi et al., 2019; Stoian et al., 2020; Wang et al., 2020). We treat ST with ASR-PT as our real baseline. We observe improved translation quality with fixed-rate subsampling, +0.47 BLEU at $k = 6$. Subsampling offers a chance to bypass noisy speech signals and reducing the number of source states makes learning translation alignment easier, but deciding the optimal sampling rate is tough. Results in Figure 5 reveal that fixed-rate subsampling deteriorates ST performance with suboptimal rates. Replacing fixed-rate subsampling with our one-layer CNN also fails to improve over the baseline, although CNN offers more flexibility in feature manipulation. By contrast to fixed-rate subsampling, the proposed AFS is data-driven, shifting the decision burden to the data and model themselves. As a result, AFS^t and AFS^{t,f} surpass ASR-PT by 0.9 BLEU and 1.71 BLEU, respectively, substantially narrowing the performance gap compared to the cascade baseline (-0.14 BLEU).

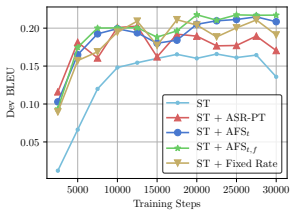


Figure 6: ST training curves (MuST-C En-De dev set). ASR pretraining significantly accelerates model convergence, and feature selection further stabilizes and improves training. $\lambda = 0.5, k = 6$.

We also observe improved decoding speed: AFS runs ∼1.37× faster than ASR-PT. Compared to the fixed-rate subsampling, AFS is slightly slower which we ascribe to the overhead introduced by the gating module. Surprisingly, Table 1 shows that the vanilla ST runs slower than ASR-PT (0.87×) while the cascade model is slightly faster (1.06×). By digging into the beam search algorithm, we discover that ASR pretraining shortens the number of steps in beam-decoding: 94 ASR-PT vs. 112 vanilla ST (on average). The speedup brought by cascading is due to the smaller English vocabulary size compared to the German vocabulary when processing audio inputs.

4.2 Why (Adaptive) Feature Selection?

Apart from the benefits in translation quality, we go deeper to study other potential impacts of (adaptive) feature selection. We begin with inspecting training curves. Figure 6 shows that ASR pretraining improves model convergence; feature selection makes training more stable. Compared to other models, the curve of ST with AFS is much smoother, suggesting its better regularization effect.

We then investigate the effect of training data size, and show the results in Figure 7. Overall, we do not observe higher data efficiency by feature selection on low-resource settings. But instead, our results suggest that feature selection delivers larger performance improvement when more training data is available. With respect to data efficiency, ASR pretraining seems to be more important (Figure 7, left) (Bansal et al., 2019; Stoian et al., 2020). Com-

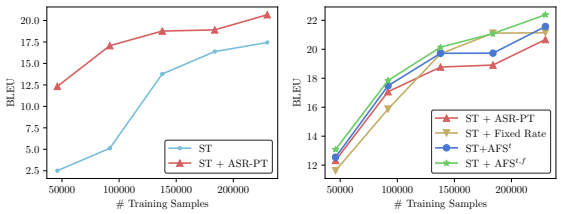


Figure 7: BLEU as a function of training data size on MuST-C En-De. We split the original training data into non-overlapped five subsets, and train different models with accumulated subsets. Results are reported on the test set. Note that we perform ASR pretraining on the original dataset. $\lambda = 0.5, k = 6$.

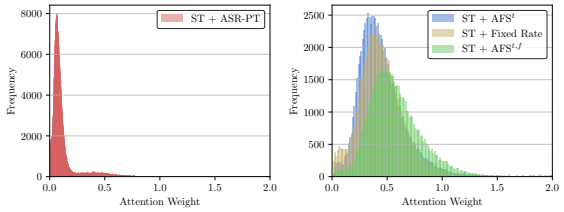
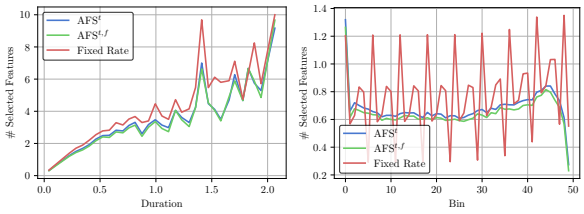


Figure 8: Histogram of the cross-attention weights received per ST encoder output on MuST-C En-De test set. For each instance, we collect attention weights averaged over different heads and decoder layers following Zhang et al. (2020). Larger weight indicates stronger impact of the encoder output on translation. Feature selection biases the distribution towards larger weights. $\lambda = 0.5, k = 6$.

pared to AFS, the fixed-rate subsampling suffers more from small-scale training: it yields worse performance than ASR-PT when data size $\leq$ 100K, highlighting better generalization of AFS.

In addition to model performance, we also look into the ST model itself, and focus on the cross-attention weights. Figure 8 visualize the attention value distribution, where ST models with feature selection noticeably shift the distribution towards larger weights. This suggests that each ST encoder output exerts greater influence on the translation. By removing redundant and noisy speech features, feature selection eases the learning of the ST encoder, and also enhances its connection strength with the ST decoder. This helps bridge the modality gap between speech and text translation. Although fixed-rate subsampling also delivers a distribution shift similar to AFS, its inferior ST performance compared to AFS corroborates the better quality of adaptively selected features.

AFS vs. Fixed Rate We compare these two approaches by analyzing the number of retained features with respect to word duration and temporal position. Results in Figure 9a show that the underlying pattern behind these two methods is similar:



(a) Duration Analysis (b) Position Analysis

Figure 9: The number of selected features vs. word duration (left) and position (right) on MuST-C En-De test set. For word duration, we align the audio and its transcription by Montreal Forced Aligner (McAuliffe et al., 2017), and collect each words' duration and its corresponding retained feature number. For position, we uniformly split each input into 50 pieces, and count the average number of retained features in each piece. $\lambda = 0.5, k = 6$.

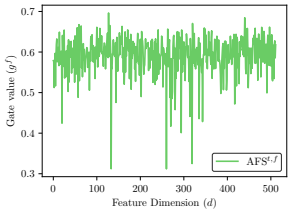


Figure 10: Illustration of feature gate $\mathbf{g}^f$ with $\lambda = 0.5$.

words with longer duration correspond to more speech features. However, when it comes to temporal position, Figure 9b illustrates their difference: fixed-rate subsampling is context-independent, periodically picking up features; while AFS decides feature selection based on context information. The curve of AFS is more smooth, indicating that features kept by AFS are more uniformly distributed across different positions, ensuring the features' informativeness.

AFS^t vs. AFS^{t,f} Their only difference lies at the feature gate $\mathbf{g}^f$. We visualize this gate in Figure 10. Although this gate induces no sparsification, it offers AFS^{t,f} the capability of adjusting the weight of each neuron. In other words, AFS^{t,f} has more freedom in manipulating speech features.

4.3 Results on MuST-C and LibriSpeech

Table 2 and Table 3 list the results on MuST-C and LibriSpeech En-Fr, respectively. Over all tasks, AFS^t/AFS^{t,f} substantially outperforms ASR-PT by 1.34/1.60 average BLEU, pruning out 84.5% temporal speech features on average and yielding an average decoding speedup of 1.45×. Our model narrows the gap against the cascade model to -0.8 average BLEU, where AFS surpasses Cascade on LibriSpeech En-Fr, without using KD (Liu et al.,

Metric	Model	De	Es	Fr	It	Nl	Pt	Ro	Ru
BLEU↑	Di Gangi et al. (2019)	17.30	20.80	26.90	16.80	18.80	20.10	16.50	10.50
	Transformer + ASR-PT*	21.77	26.41	31.56	21.46	25.22	26.84	20.53	14.31
	ST	17.44	23.85	28.43	19.54	21.23	22.55	17.66	12.10
	ST + ASR-PT	20.67	25.96	32.24	20.84	23.27	24.83	19.94	13.96
	Cascade	22.52	27.92	34.53	24.02	26.74	27.57	22.61	16.13
	ST + AFS^{t}	21.57	26.78	33.34	23.08	24.68	26.13	21.73	15.10
	ST + AFS^{t,f}	22.38	27.04	33.43	23.35	25.05	26.55	21.87	14.92
SacreBLEU ↑	ST + AFS^{t}	21.6	26.6	31.5	22.6	24.6	25.9	20.8	14.9
	ST + AFS^{t,f}	22.4	26.9	31.6	23.0	24.9	26.3	21.0	14.7
Temporal Sparsity Rate	ST + AFS^{t}	84.4%	84.5%	83.2%	84.9%	84.4%	84.4%	84.7%	84.2%
	ST + AFS^{t,f}	85.1%	84.5%	84.7%	84.9%	83.5%	85.1%	84.8%	84.7%
Speedup ↑	ST + AFS^{t}	1.38×	1.35×	1.50×	1.34×	1.54×	1.43×	1.59×	1.31×
	ST + AFS^{t,f}	1.37×	1.34×	1.50×	1.39×	1.42×	1.26×	1.46×	1.37×

Table 2: Performance over 8 languages on MuST-C dataset. *: results reported by the ESPNet toolkit (Watanabe et al., 2018), where the hyperparameters of beam search are tuned for each dataset.

Metric	Model	En-Fr
BLEU↑	Bérard et al. (2018)	13.40
	Watanabe et al. (2018)	16.68
	Liu et al. (2019a)	17.02
	Wang et al. (2019)	17.05
	Wang et al. (2020)	17.66
	ST	14.32
	ST + ASR-PT	17.05
	Cascade	18.27
	ST + AFS^{t}	18.33
	ST + AFS^{t,f}	18.56
SacreBLEU↑	ST + AFS^{t}	16.9
	ST + AFS^{t,f}	17.2
Temporal Sparsity Rate	ST + AFS^{t}	84.7%
	ST + AFS^{t,f}	83.5%
Speedup↑	ST + AFS^{t}	1.84×
	ST + AFS^{t,f}	1.78×

Table 3: Performance on LibriSpeech En-Fr.

2019a) and data augmentation (Wang et al., 2020). Comparability to previous work is limited due to possible differences in tokenization and letter case. To ease future cross-paper comparison, we provide SacreBLEU (Post, 2018)[4] for our models.

5 Related Work

Speech Translation Pioneering studies on ST used a cascade of separately trained ASR and MT systems (Ney, 1999). Despite its simplicity, this approach inevitably suffers from mistakes made by ASR models, and is error prone. Research in this direction often focuses on strategies capable of mitigating the mismatch between ASR output and MT input, such as representing ASR outputs with lattices (Saleem et al., 2004; Mathias and Byrne, 2006; Zhang et al., 2019a; Beck et al., 2019), injecting synthetic ASR errors for robust MT (Tsvetkov et al., 2014; Cheng et al., 2018) and differentiable cascade modeling (Kano et al., 2017; Anastasopoulos and Chiang, 2018; Sperber et al., 2019).

In contrast to cascading, another option is to perform direct speech-to-text translation. Duong et al. (2016) and Bérard et al. (2016) employ the attentional encoder-decoder model (Bahdanau et al., 2015) for E2E ST without accessing any intermediate transcriptions. E2E ST opens the way to bridging the modality gap directly, but it is data-hungry, sample-inefficient and often underperforms cascade models especially in low-resource settings (Bansal et al., 2018). This led researchers to explore solutions ranging from efficient neural architecture design (Karita et al., 2019; Di Gangi et al., 2019; Sung et al., 2019) to extra training signal incorporation, including multi-task learning (Weiss et al., 2017; Liu et al., 2019b), submodule pretraining (Bansal et al., 2019; Stoian et al., 2020; Wang et al., 2020), knowledge distillation (Liu et al., 2019a), meta-learning (Indurthi et al., 2019) and data augmentation (Kocabiyikoglu et al., 2018; Jia et al., 2019; Pino et al., 2019). Our work focuses on E2E ST, but we investigate feature selection which has rarely been studied before.

Speech Feature Selection Encoding speech signals is challenging as acoustic input is lengthy, noisy and redundant. To ease model learning, previous work often selected features via downsampling techniques, such as convolutional modeling (Di

[4]signature: BLEU+c.mixed+#.1+s.exp+tok.13a+version.1.3.6

Gangi et al., 2019) and fixed-rate subsampling (Lu et al., 2015). Recently, Zhang et al. (2019b) and Na et al. (2019) proposed dynamic subsampling for ASR which learns to skip uninformative features during recurrent encoding. Unfortunately, their methods are deeply embedded into recurrent networks, hard to adapt to other architectures like Transformer (Vaswani et al., 2017). Recently, Salesky et al. (2020) have explored phoneme-level representations for E2E ST, but this requires non-trivial phoneme recognition and alignment.

Instead, we resort to sparsification techniques which have achieved great success in NLP tasks recently (Correia et al., 2019; Child et al., 2019; Zhang et al., 2020). In particular, we employ $\mathcal{L}_0$DROP (Zhang et al., 2020) for AFS to dynamically retain informative speech features, which is fully differentiable and independent of concrete encoder/decoder architectures. We extend $\mathcal{L}_0$DROP by handling both temporal and feature dimensions with different gating networks, and apply it to E2E ST.

6 Conclusion and Future Work

In this paper, we propose adaptive feature selection for E2E ST to handle redundant and noisy speech signals. We insert AFS in-between the ST encoder and a pretrained, frozen ASR encoder to filter out uninformative features contributing little to ASR. We base AFS on $\mathcal{L}_0$DROP (Zhang et al., 2020), and extend it to modeling both temporal and feature dimensions. Results show that AFS improves translation quality and accelerates decoding by $\sim$1.4$\times$ with an average temporal sparsity rate of $\sim$84%. AFS successfully narrows or even closes the performance gap compared to cascading models.

While most previous work on sparsity in NLP demonstrates its benefits from efficiency and/or interpretability perspectives (Zhang et al., 2020), we show that sparsification in our scenario – E2E ST – leads to substantial performance gains.

In the future, we will work on adapting AFS to simultaneous speech translation.

Acknowledgments

We would like to thank Shucong Zhang for his great support on building our ASR baselines. IT acknowledges support of the European Research Council (ERC Starting grant 678254) and the Dutch National Science Foundation (NWO VIDI 639.022.518). This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 825460 (ELITR). Rico Sennrich acknowledges support of the Swiss National Science Foundation (MUTAMUR; no. 176727).

References

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 82–91, New Orleans, Louisiana. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2018. Low-resource speech-to-text translation. In *Proc. Interspeech 2018*, pages 1298–1302.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2019. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 58–68, Minneapolis, Minnesota. Association for Computational Linguistics.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy. Association for Computational Linguistics.

Daniel Beck, Trevor Cohn, and Gholamreza Haffari. 2019. Neural speech translation using lattice transformations and graph networks. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 26–31, Hong Kong. Association for Computational Linguistics.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-end automatic speech translation of audiobooks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6224–6228. IEEE.

Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. In *NIPS Workshop on End-to-end Learning for Speech and Audio Processing*, Barcelona, Spain.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1766, Melbourne, Australia. Association for Computational Linguistics.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Gonçalo M. Correia, Vlad Niculae, and André F. T. Martins. 2019. Adaptively sparse transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2174–2184, Hong Kong, China. Association for Computational Linguistics.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C: a Multilingual Speech Translation Corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2019. Adapting Transformer to End-to-End Spoken Language Translation. In *Proc. Interspeech 2019*, pages 1133–1137.

Long Duong, Antonios Anastasopoulos, David Chiang, Steven Bird, and Trevor Cohn. 2016. An attentional model for speech translation without transcription. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 949–959, San Diego, California. Association for Computational Linguistics.

Alex Graves, Santiago Fernández, and Faustino Gomez. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *In Proceedings of the International Conference on Machine Learning, ICML 2006*, pages 369–376.

Sathish Indurthi, Houjeung Han, Nikhil Kumar Lakumarapu, Beomseok Lee, Insoo Chung, Sangha Kim, and Chanwoo Kim. 2019. Data efficient direct speech-to-text translation with modality agnostic meta-learning. *arXiv preprint arXiv:1911.04283*.

Ye Jia, Melvin Johnson, Wolfgang Macherey, Ron J Weiss, Yuan Cao, Chung-Cheng Chiu, Naveen Ari, Stella Laurenzo, and Yonghui Wu. 2019. Leveraging weakly supervised data to improve end-to-end speech-to-text translation. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7180–7184. IEEE.

Takatomo Kano, Sakriani Sakti, and Satoshi Nakamura. 2017. Structured-based curriculum learning for end-to-end english-japanese speech translation. In *Proc. Interspeech 2017*, pages 2630–2634.

Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, Shinji Watanabe, Takenori Yoshimura, and Wangyou Zhang. 2019. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 449–456.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Ali Can Kocabiyikoglu, Laurent Besacier, and Olivier Kraif. 2018. Augmenting librispeech with French translations: A multimodal corpus for direct speech translation evaluation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Yuchen Liu, Hao Xiong, Jiajun Zhang, Zhongjun He, Hua Wu, Haifeng Wang, and Chengqing Zong. 2019a. End-to-End Speech Translation with Knowledge Distillation. In *Proc. Interspeech 2019*, pages 1128–1132.

Yuchen Liu, Jiajun Zhang, Hao Xiong, Long Zhou, Zhongjun He, Hua Wu, Haifeng Wang, and Chengqing Zong. 2019b. Synchronous speech recognition and speech-to-text translation with interactive decoding. *arXiv preprint arXiv:1912.07240*.

Christos Louizos, Max Welling, and Diederik P. Kingma. 2018. Learning sparse neural networks through l_0 regularization. In *International Conference on Learning Representations*.

Liang Lu, Xingxing Zhang, Kyunghyun Cho, and Steve Renals. 2015. A study of the recurrent neural network encoder-decoder for large vocabulary speech recognition. In *Sixteenth Annual Conference of the International Speech Communication Association*.

Lambert Mathias and William Byrne. 2006. Statistical phrase-based speech translation. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pages I–I. IEEE.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Proc. Interspeech 2017*, pages 498–502.

Rui Na, Junfeng Hou, Wu Guo, Yan Song, and Lirong Dai. 2019. Learning adaptive downsampling encoding for online end-to-end speech recognition. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 850–854.

Hermann Ney. 1999. Speech translation: Coupling of recognition and translation. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, volume 1, pages 517–520. IEEE.

Jan Niehues, Roldano Cattoni, Sebastian Stüker, Matteo Negri, Marco Turchi, Elizabeth Salesky, Ramon Sanabria, Loïc Barrault, Lucia Specia, and Marcello Federico. 2019. The iwslt 2019 evaluation campaign. In *Proceedings of the 16th International Workshop on Spoken Language Translation (IWSLT 2019)*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Juan Pino, Liezl Puzon, Jiatao Gu, Xutai Ma, Arya D McCarthy, and Deepak Gopinath. 2019. Harnessing indirect training data for end-to-end automatic speech translation: Tricks of the trade. In *Proceedings of the 16th International Workshop on Spoken Language Translation (IWSLT)*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Shirin Saleem, Szu-Chen (Stan) Jou, Stephan Vogel, and Tanja Schultz. 2004. Using word lattice information for a tighter coupling in speech translation systems. In *International Conference of Spoken Language Processing*.

Elizabeth Salesky, Matthias Sperber, and Alan W. Black. 2020. Exploring phoneme-level speech representations for end-to-end speech translation. In *Proceedings of the 2020 Annual Conference of the Association for Computational Linguistics*. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2019. Attention-passing models for robust and data-efficient end-to-end speech translation. *Transactions of the Association for Computational Linguistics*, 7:313–325.

Mihaela C Stoian, Sameer Bansal, and Sharon Goldwater. 2020. Analyzing asr pretraining for low-resource speech-to-text translation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7909–7913. IEEE.

Tzu-Wei Sung, Jun-You Liu, Hung-yi Lee, and Lin-shan Lee. 2019. Towards end-to-end speech-to-text translation with two-pass decoding. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7175–7179. IEEE.

Yulia Tsvetkov, Florian Metze, and Chris Dyer. 2014. Augmenting translation models with simulated acoustic confusions for improved spoken language translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 616–625, Gothenburg, Sweden. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Chengyi Wang, Yu Wu, Shujie Liu, Zhenglu Yang, and Ming Zhou. 2019. Bridging the gap between pretraining and fine-tuning for end-to-end speech translation. *arXiv preprint arXiv:1909.07575*.

Chengyi Wang, Yu Wu, Shujie Liu, Ming Zhou, and Zhenglu Yang. 2020. Curriculum pre-training for end-to-end speech translation. *arXiv preprint arXiv:2004.10093*.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. Espnet: End-to-end speech processing toolkit. In *Proc. Interspeech 2018*, pages 2207–2211.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-sequence models can directly translate foreign speech. In *Proc. Interspeech 2017*, pages 2625–2629.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2020. On sparsifying encoder outputs in sequence-to-sequence models. *arXiv preprint arXiv:2004.11854*.

Pei Zhang, Niyu Ge, Boxing Chen, and Kai Fan. 2019a. Lattice transformer for speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6475–6484, Florence, Italy. Association for Computational Linguistics.

Shucong Zhang, Erfan Loweimi, Yumo Xu, Peter Bell, and Steve Renals. 2019b. Trainable Dynamic Subsampling for End-to-End Speech Recognition. In *Proc. Interspeech 2019*, pages 1413–1417.